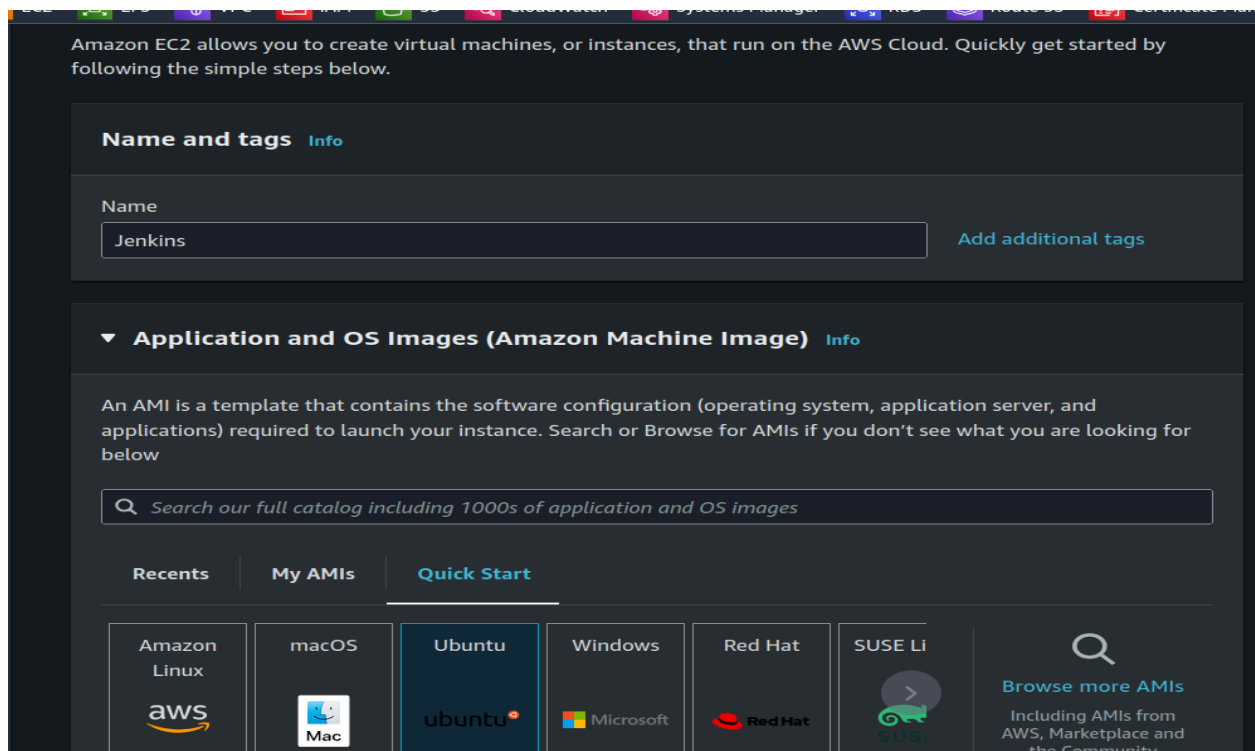
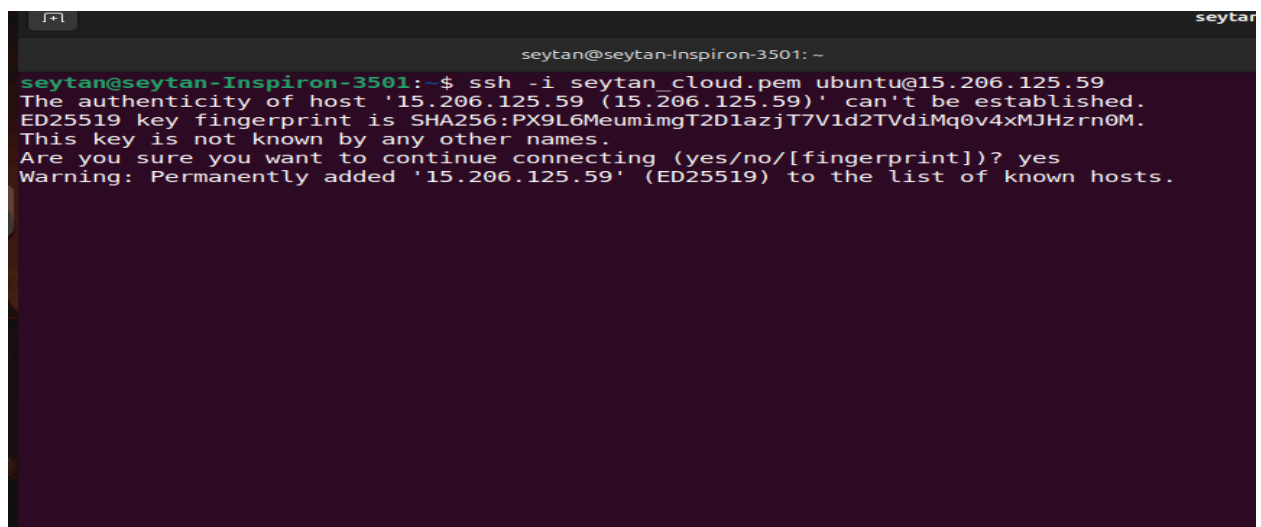


Task : Jenkins-CI_CD-Pipeline-for-Insure-Me-Project

Step 1: Create and Launch an ec2 Instance.



Take ssh of it → `ssh -i private.pem user@public_ip`



Step 2: Install the Jenkins tool.

Commands

→ **sudo apt update -y**
sudo apt install fontconfig openjdk-17-jre -y

**sudo wget -O /usr/share/keyrings/jenkins-keyring.asc **
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo deb

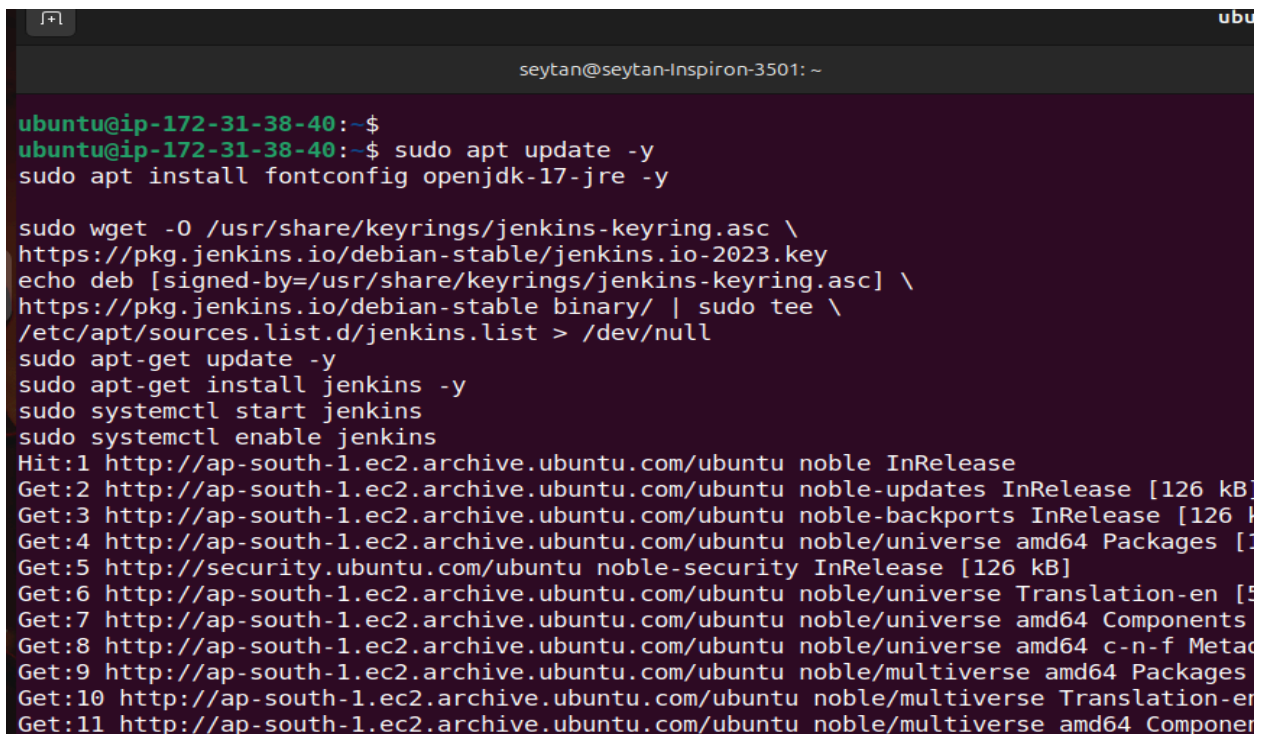
**[signed-by=/usr/share/keyrings/jenkins-keyring.asc] **
**https://pkg.jenkins.io/debian-stable binary/ | sudo tee **
/etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update -y

sudo apt-get install jenkins -y

sudo systemctl start jenkins

sudo systemctl enable jenkins

A terminal window screenshot showing the execution of commands to install Jenkins. The terminal title is 'seytan@seytan-Inspiron-3501: ~'. The commands and their outputs are as follows:
1. `ubuntu@ip-172-31-38-40:~$`
2. `ubuntu@ip-172-31-38-40:~$ sudo apt update -y`
3. `sudo apt install fontconfig openjdk-17-jre -y`
4. `sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \`
5. `https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key`
6. `echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \`
7. `https://pkg.jenkins.io/debian-stable binary/ | sudo tee \`
8. `/etc/apt/sources.list.d/jenkins.list > /dev/null`
9. `sudo apt-get update -y`
10. `sudo apt-get install jenkins -y`
11. `sudo systemctl start jenkins`
12. `sudo systemctl enable jenkins`
13. `Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease`
14. `Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]`
15. `Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]`
16. `Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [126 kB]`
17. `Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]`
18. `Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [512 B]`
19. `Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [126 kB]`
20. `Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [126 kB]`
21. `Get:9 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [126 kB]`
22. `Get:10 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [126 kB]`
23. `Get:11 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [126 kB]`

Step 3: Install Docker and Maven .(reuiired to build image and container)

Commands→

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

Add Docker's official GPG key:

```
sudo apt-get update -y
```

```
sudo apt-get install ca-certificates curl -y
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
```

```
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

Add the repository to Apt sources:

```
echo \
```

```
"deb [arch=$(dpkg --print-architecture)
```

```
signed-by=/etc/apt/keyrings/docker.asc]
```

```
https://download.docker.com/linux/ubuntu \
```

```
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
```

```
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt-get update -y
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

```
docker-buildx-plugin docker-compose-plugin -y
```

```
sudo docker --version
```

```
Sudo apt install maven -y
```

```
Sudo usermod -aG docker jenkins
```

```
Sudo systemctl restart jenkins
```

```
ubuntu@ip-172-31-38-40:~$ for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

```
# Add Docker's official GPG key:
```

```
sudo apt-get update -y
```

```
sudo apt-get install ca-certificates curl -y
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
```

```
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

```
# Add the repository to Apt sources:
```

```
echo \
```

```
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
```

```
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
```

```
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt-get update -y
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
```

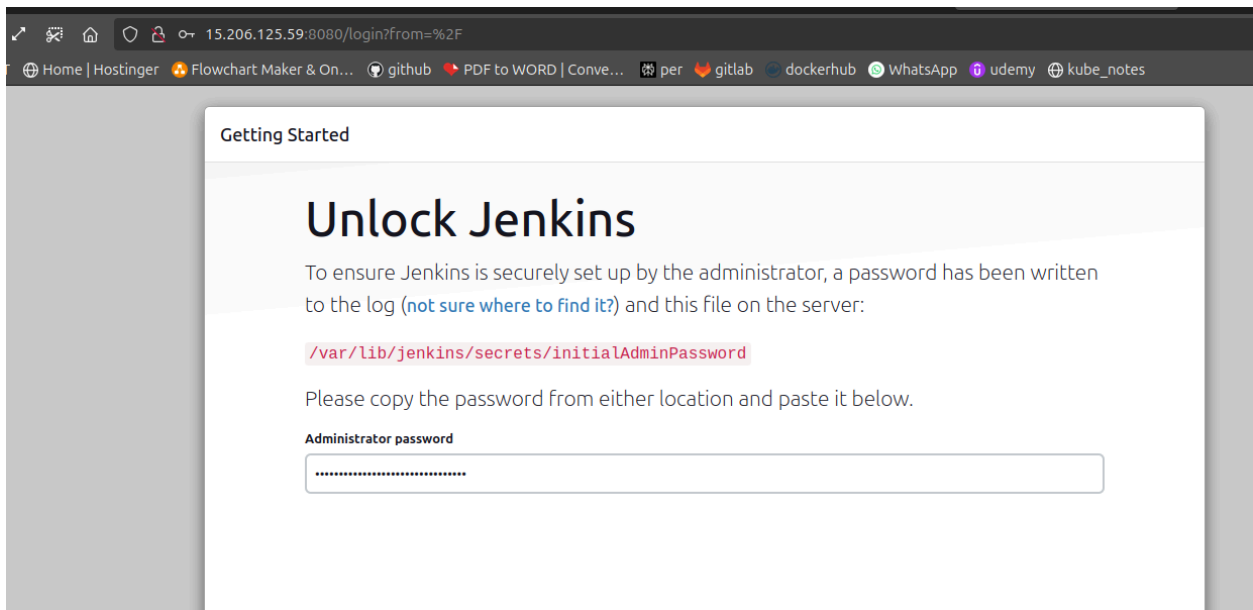
```
sudo docker --version
```

```
seytan@seytan-Inspiron-3501: ~  
ubuntu@ip-172-31-38-40:~$ sudo systemctl restart jenkins
```

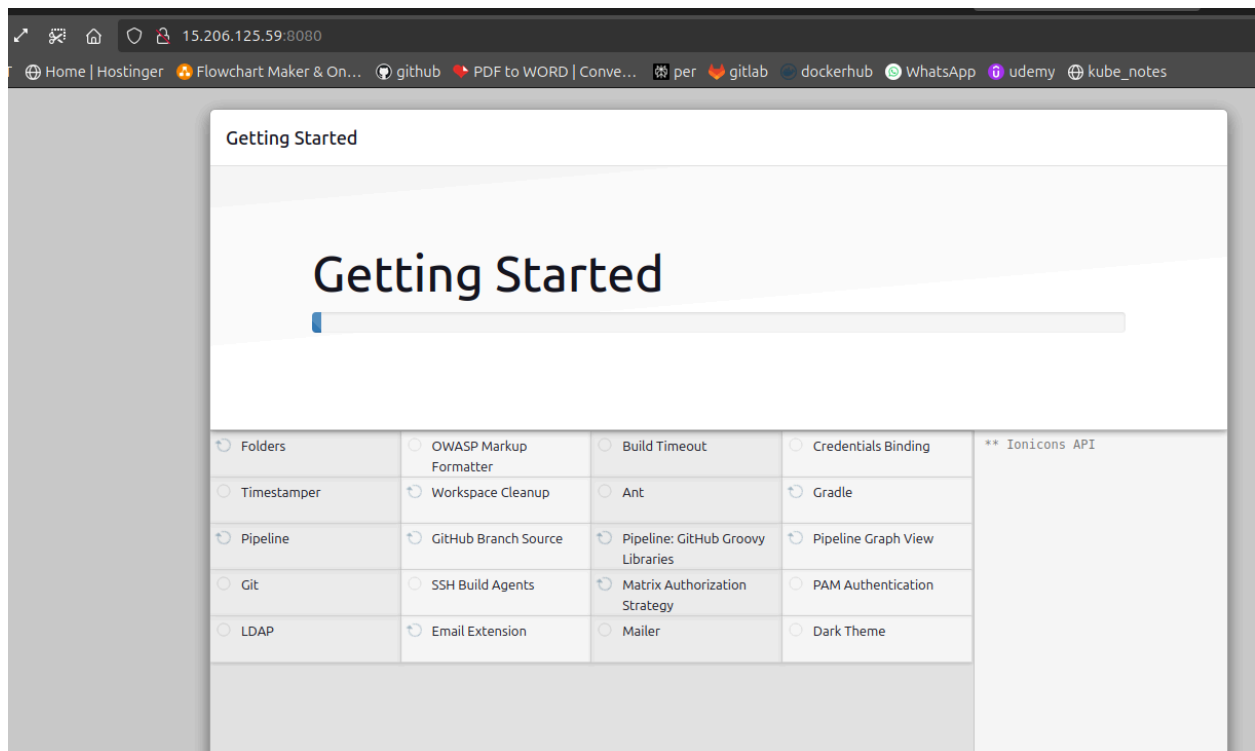
Step 4: Now Navigate to the `/var/lib/jenkins/secrets` folder and copy the password from `initialAdminpassword` file.

```
root@ip-172-31-38-40:~# cd /var/lib/jenkins/secrets/  
root@ip-172-31-38-40:/var/lib/jenkins/secrets# ls  
initialAdminPassword jenkins.model.Jenkins.crumbSalt master.key  
root@ip-172-31-38-40:/var/lib/jenkins/secrets# cat initialAdminPassword  
f59ff5930a01457abcf143a20321b852  
root@ip-172-31-38-40:/var/lib/jenkins/secrets# _
```

After this paste the public ip of instance in the web-browser and port 8080 which is default port of jenkins.



Step 5: Install the default plugins.



Step 6: After installing plugins it will ask to create admin user enter the details and continue.

The screenshot shows the Jenkins 'Create First Admin User' form. The form is titled 'Create First Admin User' and is located on the 'Getting Started' page. The form has five input fields: 'Username' (gaurav), 'Password' (•), 'Confirm password' (•), 'Full name' (gaurav), and 'E-mail address' (admin@gmail.com). The 'E-mail address' field is highlighted with a blue border. At the bottom of the form, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

Getting Started

Create First Admin User

Username
gaurav

Password
•

Confirm password
•

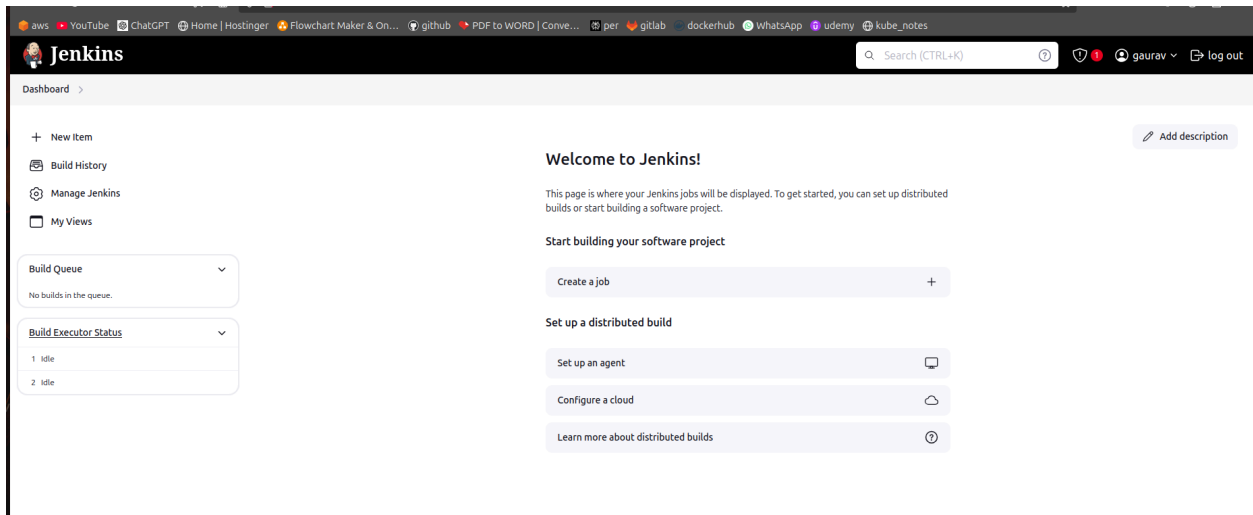
Full name
gaurav

E-mail address
admin@gmail.com

Jenkins 2.462.3

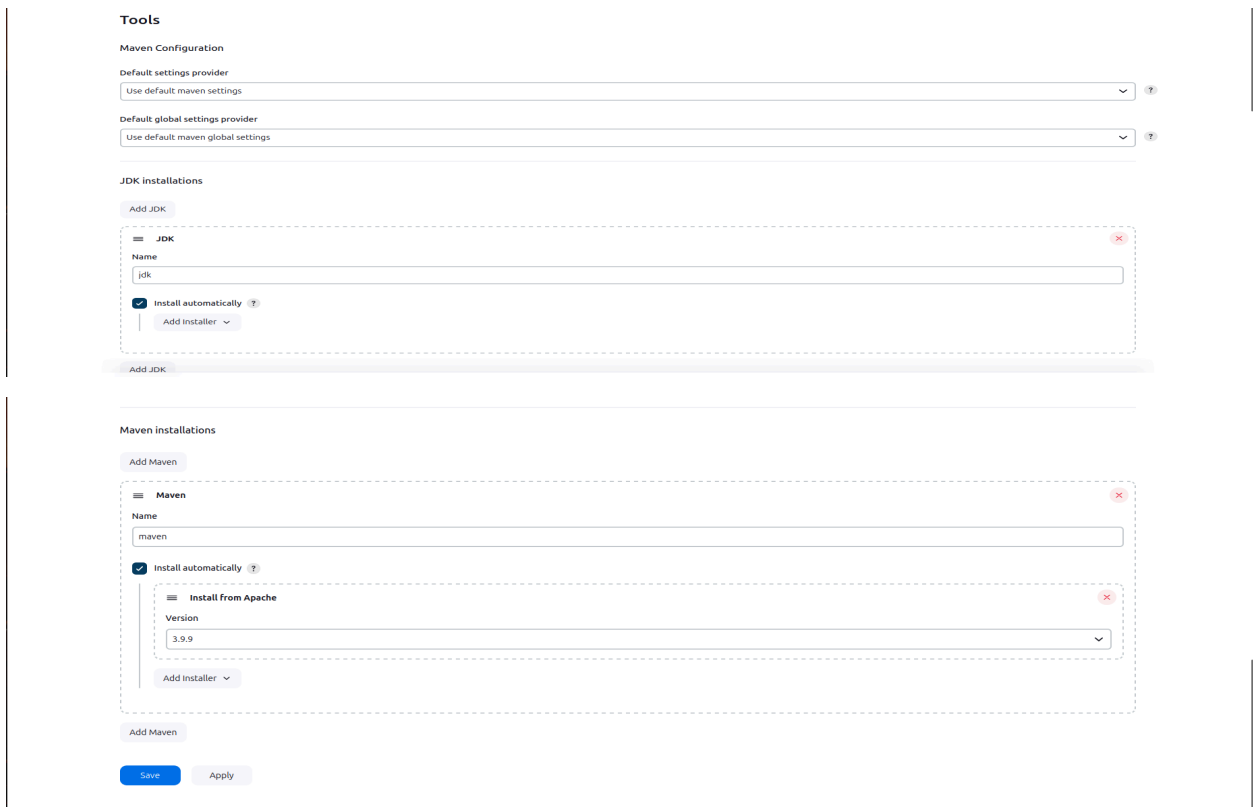
[Skip and continue as admin](#) [Save and Continue](#)

Output :



Step 7: Install the essential plugins from the Manage Jenkins tab .

After this we can configure the plugins in the Tools section.





Step 8: Now click on new job and create a pipeline .


New Item


Enter an item name


Select an item type


**Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

Step 9: After this go to the configure section and write script for our pipeline.

Pipeline script

Script ?

```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Code-Checkout') {
6       steps {
7         git branch: 'main', changelog: false, poll: false, url: 'https://github.com/Gaurav1251/Jenkins_Tasks'
8       }
9     }
10    stage('Code-Build') {
11      steps {
12        sh 'cd Jenkins-Task-Integrate-GitHub-and-Build-Insure-Me-Project/ && mvn clean package'
13      }
14    }
15    stage('Containerize the application'){
16      steps {
17        echo 'Creating Docker image'
18      }
19    }
20  }
21 }
```

☒ Use Groovy Sandbox ?

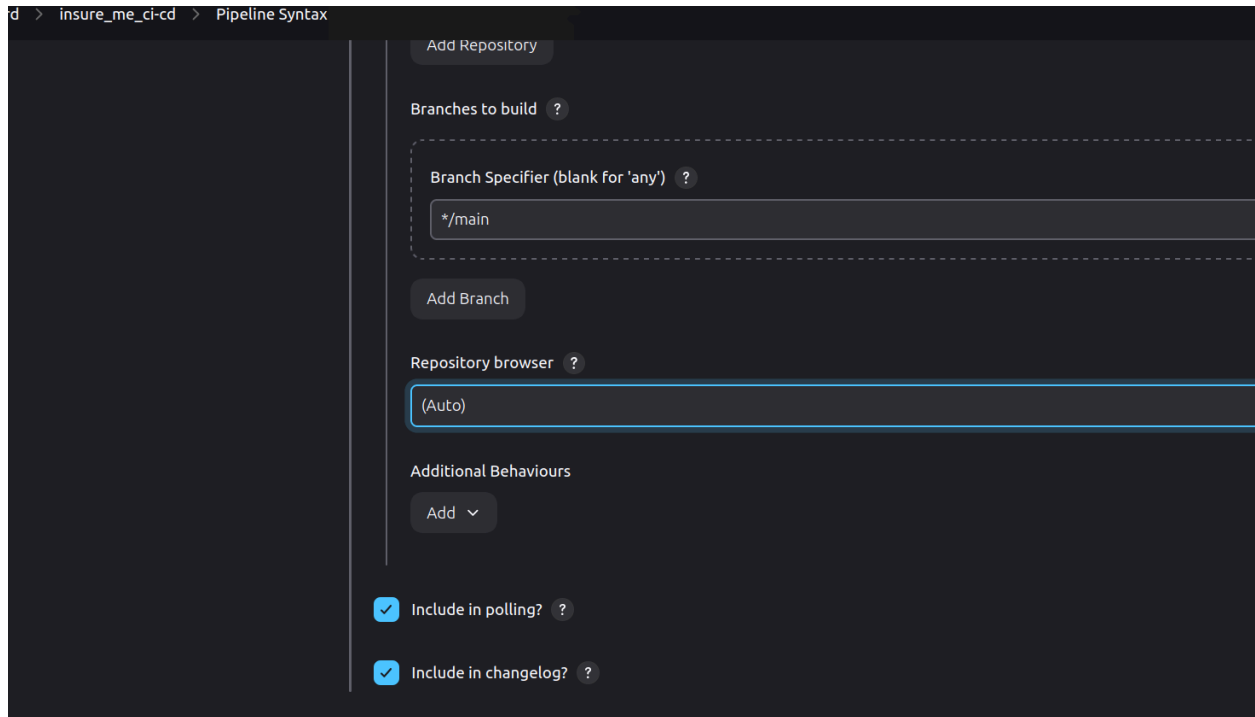
[Pipeline Syntax](#)

Save

Apply

REST API Jenkins 2.462.3

**Now for the checkout stage we require a github repo.
So we will generate a Pipeline syntax for it using the Pipeline Syntax button.**



The screenshot shows a web interface for configuring a pipeline stage. The breadcrumb navigation at the top reads "d > insure_me_ci-cd > Pipeline Syntax". The main configuration area is on the right, with a left sidebar that is currently empty. The configuration options include:

- Add Repository** button
- Branches to build** section with a help icon (?), containing a dashed box and a **Branch Specifier (blank for 'any')** section with a help icon (?). Below this is a text input field containing `*/main`.
- Add Branch** button
- Repository browser** section with a help icon (?), containing a dropdown menu currently set to **(Auto)**.
- Additional Behaviours** section with an **Add** button and a dropdown arrow.
- Two checked checkboxes at the bottom: **Include in polling?** and **Include in changelog?**, each with a help icon (?).

Here we select source as checkout from version control and give the repo url and branch name from where we want to clone the code.

If repo is Private we need to provide credentials.

Now apply and save the script .

Configure

General

Advanced Project Options

Pipeline

Definition

Pipeline script

Script ?

```

1 pipeline {
2   agent any
3   stages {
4     stage('Code-Checkout') {
5       steps {
6         git branch: 'main', changelog: false, poll: false, url: 'https://github.com/Gaurav1251/Insure
7       }
8     }
9   }
10  }
11  stage('Code-Build') {
12    steps {
13      sh 'mvn clean package'
14    }
15  }
16  stage('Containerize the application'){
17  }
18  }
19 }

```

try sample Pipeline...

☒ Use Groovy Sandbox ?

Pipeline Syntax

Step 10: Now we need the docker push script so we have to generate one more pipeline syntax for logging to dockerhub .

For that 1st create credentials.

insure_the_code Pipeline Syntax

Jenkins Credentials Provider: Jenkins

Domain

Global credentials (unrestricted)

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

Gaurav1251

☒ Treat username as secret ?

Password ?

.....

ID ?

After creating credentials , go to pipeline syntax and select withCredential:bind credentials to variables.

Give username and password variables and generate the script.

The screenshot shows a configuration window titled "Sample Step" for the step "withCredentials: Bind credentials to variables". Below the title bar, there is a section for "withCredentials" with a help icon. A note states: "Secret values are masked on a best-effort basis to prevent *accidental* disclosure. Multiline secrets, such as the contents of a SSH private key file, are not masked. See the inline help for details and usage guidelines." Below this is a "Bindings" section containing a dashed box with a title "Username and password (separated)" and a close button. Inside this box, there are three fields: "Username Variable" with the value "docker_u", "Password Variable" with the value "docker_p", and "Credentials" with a dropdown menu showing "docker_push". At the bottom of the dashed box is a "+ Add" button.

Sample Step

withCredentials: Bind credentials to variables

withCredentials ?

Secret values are masked on a best-effort basis to prevent *accidental* disclosure. Multiline secrets, such as the contents of a SSH private key file, are not masked. See the inline help for details and usage guidelines.

Bindings

Username and password (separated) ?

Username Variable ?

docker_u

Password Variable ?

docker_p

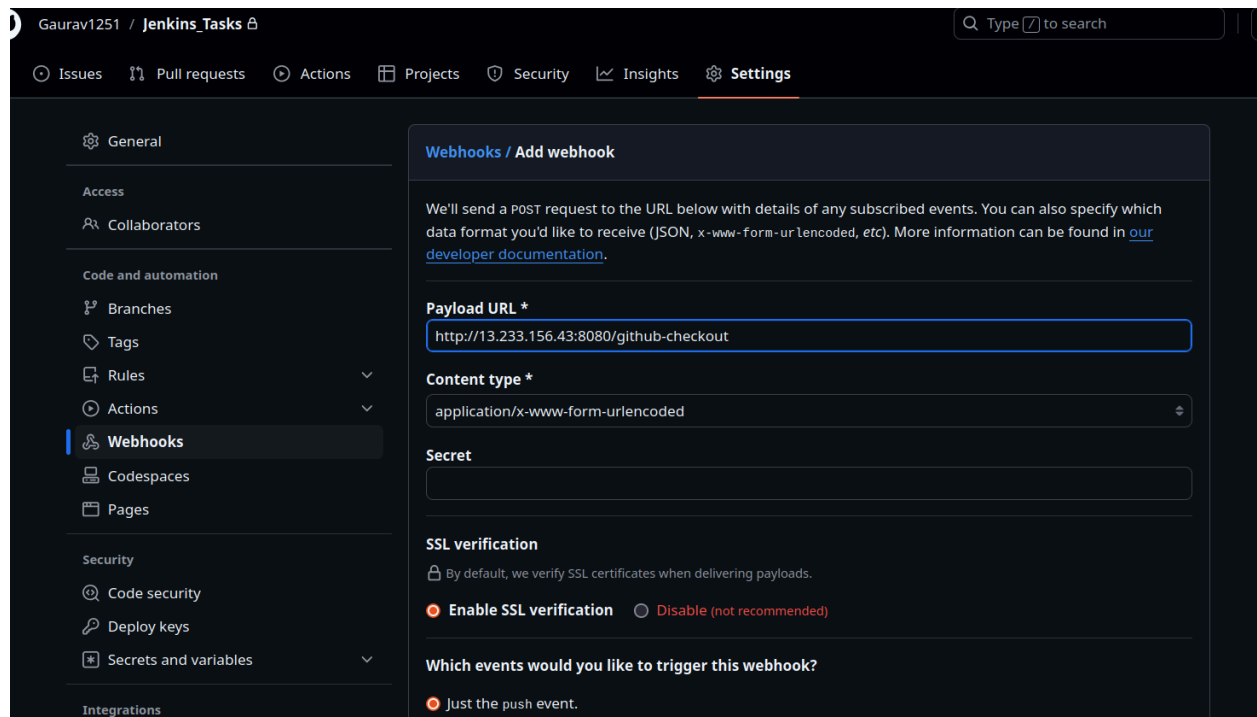
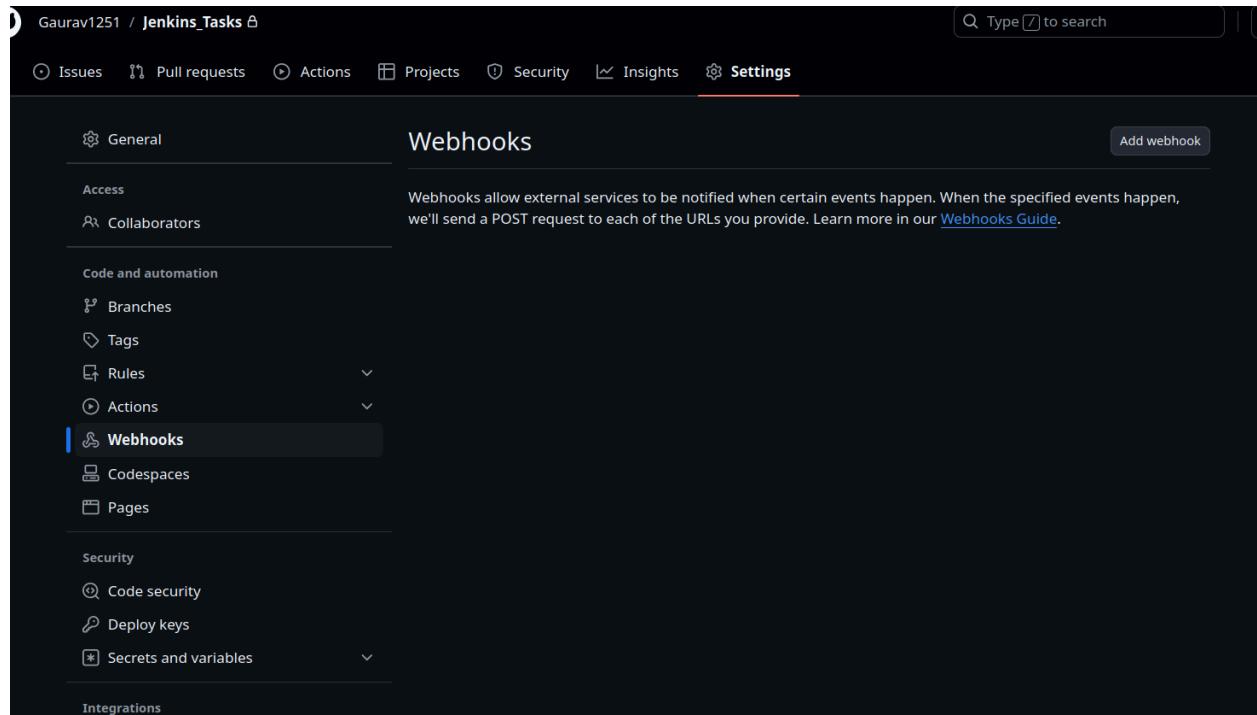
Credentials ?

docker_push


+ Add


Step 11: Paste this script in the push stage .


Now go to github repo and create a webhook pointing to jenkins url.





Step 12 : Save and Build the pipeline script.


 Status


 Changes


 Build Now


 Configure


 Delete Pipeline

 Stages



 Rename



 Pipeline Syntax


 Build History trend ▾

 Filter...

/

 #2	<u>Oct 23, 2024, 4:50 PM</u>
 #1	<u>Oct 23, 2024, 4:49 PM</u>

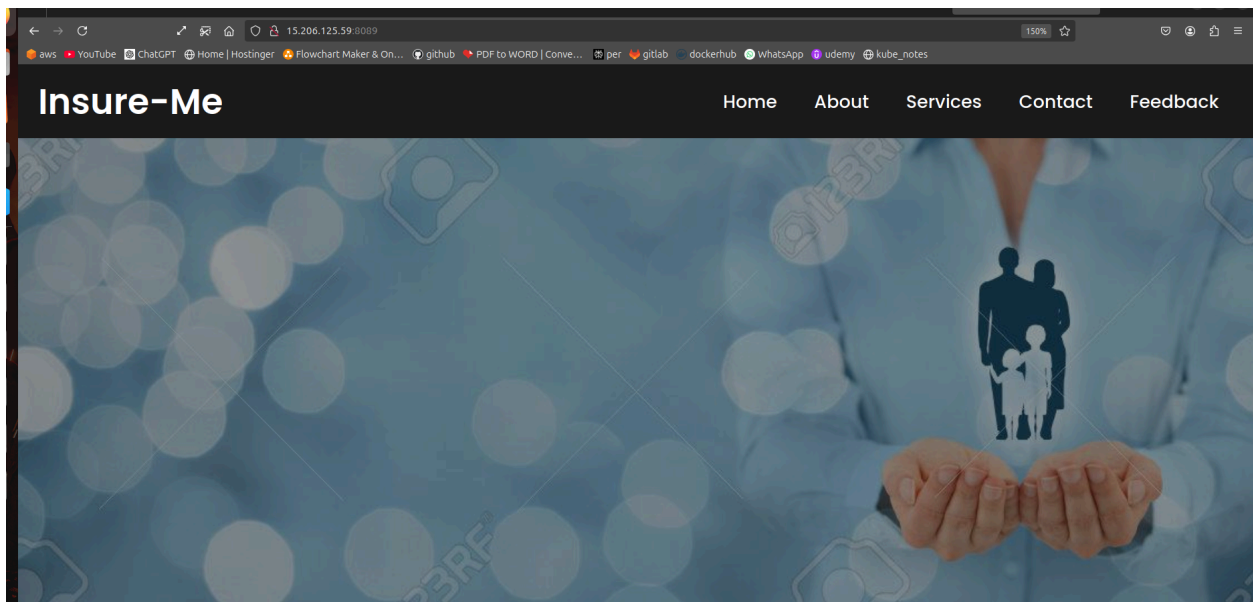
 Atom feed for all  Atom feed for failures


Pe

Step 13: Now go to the instance and check that our images are build and deployed.

```
seylan@seylan-inspiron-3501: ~
root@ip-172-31-38-40:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS    NAMES
adcf6a3d4eb   insure   "java -jar /app.jar"    About a minute ago    Up About a minute    8089/tcp, 0.0.0.0:8089->8081/tcp, [::]:8089->8081/tcp    elastic_jang

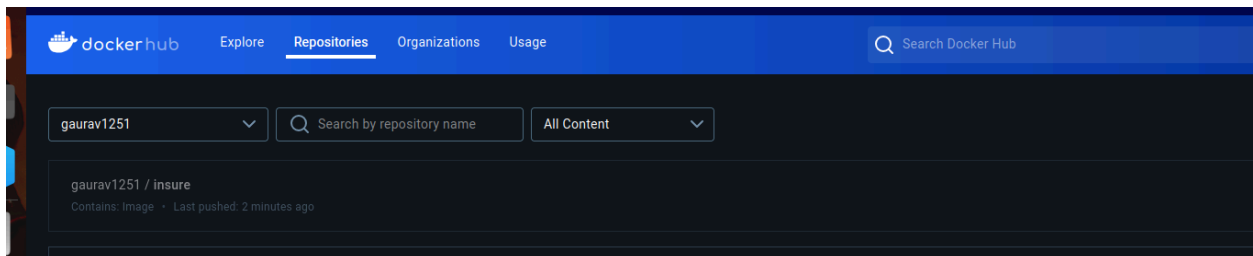
root@ip-172-31-38-40:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
insure        latest   71240c91c7d9   About a minute ago    706MB
root@ip-172-31-38-40:~#
```



Step 14: Now make some changes to repo and you will see that our pipeline will be triggered and the application will be re-build and re-deployed again.

Output:

Image pushed to dockerhub.



13.233.156.43:8080/job/insure_me_ci-cd/GitHubHookLog/

aws YouTube ChatGPT Home | Hostinger Flowchart Maker & On... github PDF to WORD | Conve... per gitlab dockerhub WhatsApp udemu kube_notes

Jenkins

Search (CTRL+K)

admin log out

Dashboard > insure_me_ci-cd > GitHub Hook Log

Status

Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Stages

Rename

Pipeline Syntax

GitHub Hook Log

Last GitHub Push

```
Started on Oct 24, 2024, 10:25:58 AM
Started by event from 140.82.115.44 → http://13.233.156.43:8080/github-webhook/ on Thu Oct 24 10:25:58 UTC 2024
Using strategy: Default
[poll] Last Built Revision: Revision 9273bb21f684862d9403a3ca9b418b44abb122f1 (refs/remotes/origin/main)
The recommended git tool is: NONE
using credential cicd
> git --version # timeout=10
> git --version # 'git version 2.43.0'
using GIT_ASKPASS to set credentials
> git ls-remote -h -- https://github.com/Gaurav1251/Jenkins_Tasks # timeout=10
Found 1 remote heads on https://github.com/Gaurav1251/Jenkins_Tasks
[poll] Latest remote head revision on refs/heads/main is: 8fcb2ead20c1ca6fc95903bd6092b2de6a711f4b
Done. Took 0.76 sec
Changes found
```

Build History trend

13.233.156.43:8080/#

aws YouTube ChatGPT Home | Hostinger Flowchart Maker & On... github PDF to WORD | Conve... per gitlab dockerhub WhatsApp udemu kube_notes

Insure-Me

Home About Services Contact Feedback